INFOGRAPHIE

Daniel Thalmann

Ecole Polytechnique Fédérale de Lausanne

Mars 2003

1. LES CONCEPTS DE BASE

1.1 Le rôle de l'informatique graphique et ses applications

1.1.1 Introduction

Il y a déjà 35 ans que les premiers dessins par ordinateur ont été produits notamment pour le système de défense et de contrôle aérien SAGE et au M.I.T. avec l'ordinateur TX1. Pendant plus de 25 ans, certains scientifiques ont utilisé la capacité des ordinateurs pour produire des diagrammes et des graphes dans leurs rapports, thèses et articles. Pendant toute cette période, le public, même le public averti tel que la communauté universitaire, ignorait littéralement les possibilités graphiques de l'ordinateur. Il faut tout de même convenir que le matériel était encore cher, pas toujours très maniable et les résultats pas toujours très spectaculaires.

Aujourd'hui, la situation a radicalement changé, tous les micro-ordinateurs personnels ont des capacités graphiques. Les millions de télespectateurs des pays occidentaux sont envahis par les génériques et logos produits par ordinateur. On est d'ailleurs parfois ébahi par le réalisme et la perfection de certaines images générées par ordinateur. Que ce soit dans le domaine technique, médical ou simplement artistique, ces images forcent notre admiration et nous questionnent. Dans le domaine de la synthèse d'images réalistes, en moins de dix ans, on a assisté à des progrès spectaculaires, autant du point de vue matériel que du point de vue logiciel. Il faut d'ailleurs remarquer que leur évolution est intimement liée. Il serait difficilement concevable de produire des images réalistes sans disposer d'un terminal graphique de haute résolution avec au moins quelques centaines de couleurs affichables simultanément.

Plus récemment, c'est le domaine du multimédia qui s'est développé grâce aux progrès fulgurant dans les télécommunications. Avec Internet et World Wide Web, on peut consulter ou échanger des images créées à l'autre bout du monde. Par la réalité virtuelle, on peut plonger aussi dans des mondes virtuels et de nouveau, grâce aux télécommunications rapides comme les réseaux ATM, il est même possible de partager les mondes virtuels entre des participants du monde entier.

1.1.2 Les applications de l'informatique graphique

L'informatique graphique trouve ses applications dans presque tous les domaines de la création, de la conception, de la fabrication, de l'information.

En architecture, on peut représenter des bâtiments avec tous leurs détails; on peut visionner ces bâtiments avec n'importe quel point de vue.

En cartographie, l'ordinateur peut produire très rapidement des cartes avec différents types d'éléments géographiques. Dans une carte météorologique, par exemple, on peut utiliser des symboles comme des nuages ou le soleil pour indiquer quel temps il va faire dans chaque région. Notons que ce type d'application est utilisée par la télévision et dans le cadre des système vidéotex.

Dans le domaine des transports, des cartes de réseaux de bus, de chemins de fer ou d'avions peuvent être produites par ordinateur, aussi bien pour les réseaux existants que pour la planification de nouveaux réseaux.

L'ordinateur joue un rôle de plus en plus important dans la conception et la fabrication industrielle. La conception assistée par ordinateur (CAO) est particulièrement développée dans le domaine de l'automobile, de l'aéronautique, de la mécanique et de l'électronique. L'ordinateur joue un rôle fondamental puisqu'il permet de calculer des formes graphiques à trois dimensions en un temps très court. Ainsi le concepteur peut faire varier interactivement des paramètres et voir immédiatement le résultat visuel de son action. Etant donné la diversité et la complexité des formes graphiques et des structures pouvant intervenir dans les dessins de CAO, des bases de données graphiques sont nécessaires pour stocker cette masse d'informations. L'intégration de la CAO et de la fabrication assistée par ordinateur (FAO) prend de plus en plus d'essor. Par exemple, dans le domaine du textile, on peut, par CAO, concevoir les motifs d'un tissu et ceux-ci sont transmis directement aux machines produisant les textiles.

Dans les domaines de pointe tels que la bureautique et la télématique, l'édition graphique joue un rôle de plus en plus important. L'usager conçoit des dessins, les transforme et les conserve pour un usage ultérieur. Ces dessins peuvent servir aussi bien à illustrer des livres ou des textes (*celui que vous avez en mains en est un exemple typique!!!*) qu'à être transmis à des milliers de personnes via les réseaux de télécommunications (Internet, ATM).

Il ne faut évidemment pas oublier tous les dessins illustrant des données numériques, tels que les courbes, histogrammes et autres diagrammes qui servent aussi bien au scientifique pour comprendre les conséquences de ses expériences qu'au gestionnaire pour prendre des décisions.

En médecine, de nombreuses applications existent. Grâce au traitement d'images, on peut facilement détecter certaines maladies. Par exemple, en cardiologie, on peut déceler l'infarctus du myocarde par reconstruction d'images. Et on commence déjà à avoir des modèles humains entièrement synthétisés pour simuler des expériences.

Un aspect qui connaît un essor considérable est l'animation par ordinateur. Dans ce domaine, deux approches sont à considérer: l'animation assistée par ordinateur et l'animation modélisée. Dans la première approche, l'ordinateur est utilisé pour améliorer le rendement dans la production de dessins animés traditionnels. En particulier, la tâche ingrate de production de dessins intermédiaires est automatisée. Dans l'animation modélisée, les objets synthétisés évoluent dans l'espace tridimensionnel et ce nouveau type d'animation spectaculaire est visible dans des effets spéciaux au cinéma et à la TV.

Finalement, dans le domaine des arts autres que l'animation, l'ordinateur a encore de la peine à acquérir ses lettres de noblesse. Poutant, des peintures et des sculptures sont produites chaque année et des expositions font de plus en plus leur apparition.

La Figure 1-1 nous montre un tableau des domaines d'application de l'informatique graphique et quelles sont les professions touchées par cette nouvelle technologie.

CAO		
-circuits VLSI	électroniciens	- architecture
	architectes	
-génie civil	ingénieurs-électroniciens	
-construction navale	constructeurs de bateaux	
Applications biomédicales		
-aide au diagnostic	médecins	
-radiologie et médecine nucléaire		
-chirurgie plastique		
-simulation d'articulations et de mouvements	orthopédistes	- médecine nucléaire
-simulation d'opération	chirurgiens	
Visualisation scientifique		
-représentations atomiques et moléculaires	chimistes	
-simulation de phénomènes physiques	physiciens, ingénieurs	- simulation de plantes, d'animaux
	biologistes	
Graphiques en gestion		
-bureautique	employés de bureau	
-prédiction de marché	administrateurs	
-statistiques	statisticiens	
-système d'information	presse	
-analyse financière	financiers	
-éditeurs de graphiques	imprimeurs, éditeurs	
Apprentissage et éducation		
-applications pédagogiques	enseignants	
-didacticiels	étudiants	
Cartographie		
-planification urbaine	urbanistes	
-cartes géographiques	cartographes, géographes	
-cartes géologiques	géologues	
-systèmes d'information géographiques	géographes	
-météorologie	météorologues	
Technologie du vidéo		
-vidéotex	informateurs	- téléconférence
	conférenciers	- résultats d'élections
	réalisateurs TV	
-sondages	commentateurs	
Arts visuels		
-conception graphique	designers, graphistes	- conception de produits industriels
	industriels	
-architecture d'intérieur	architectes d'intérieur	- peinture et décoration
	peintres, décorateurs	
Multimedia et audiovisuel		
-production d'images réalistes	publicitaires, designers	
-jeux et divertissements	grand public	
-animation	animateurs, cinéastes	- publicité TV et journaux
	publicitaires	
-simulation de vols aériens	pilotes, contrôleurs	
-architecture, paysages	architectes, paysagistes	

Figure 1-1. Domaine d'application de l'infographie et impact sur les professions

1.2 L'architecture matériel et logiciel des systèmes graphiques

1.2.1 Les composantes des systèmes graphiques

On appelle système graphique, un ensemble de composantes matérielles et logicielles permettant de produire des dessins et des images par ordinateur. Un tel système doit permettre l'entrée, le traitement et la sortie d'informations de nature graphique. Il peut être représenté par un modèle comme celui de la Figure 1-2. Un modèle de système graphique où les parties matérielles et logicielles ne sont pas indiquées.

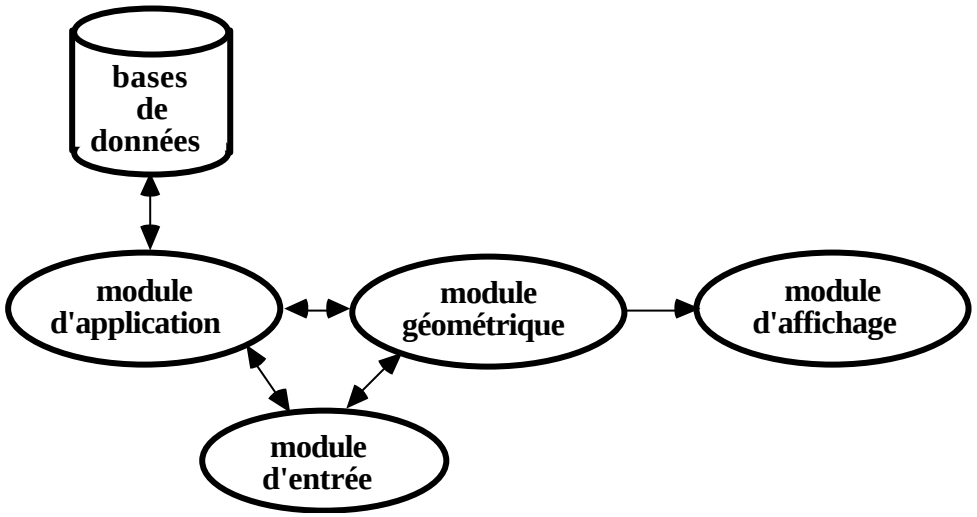

Figure 1-2. Un modèle de système graphique

Dans ce modèle, on distingue quatre modules principaux:

1. le module d'entrée: il est responsable de l'entrée des dessins et comprend normalement des dispositifs d'entrée tels qu'une tablette graphique, par exemple, et le logiciel de contrôle de ces dispositifs.

2. le module géométrique: il est formé essentiellement de logiciel et son rôle est de créer et de manipuler des objets graphiques; il faut noter que les stations graphiques les plus récentes offrent de plus en plus de facilités matérielles pour créer et manipuler des objets graphiques: rotations matérielles, générateur de cercles.

3. le module d'affichage: il est responsable de la sortie des dessins et comprend le matériel de sortie, tel que les écrans, les imprimantess ainsi que le logiciel d'affichage qui peut aller du simple ensemble de sous-programmes de traçage de lignes jusqu'au logiciel super sophistiqué de synthèse d'images avec multiples sources de lumière, ombre portée, transparence.

4. le module d'application: ce module est celui orienté vers l'usager; il diffère évidemment suivant le type d'application et se présente généralement comme un programme interactif.

Ce modèle est évidemment théorique et pratiquement, on peut plutôt considéré qu'on a un ensemble de dispositifs matériels d'entrée et de sortie, un logiciel graphique de base et un logiciel d'application. Le logiciel graphique de base comprend généralement des opérations d'entrée, de sortie et de création et manipulation d'objets graphiques.

1.2.2 Les dispositifs d'entrée graphique

Ces dispositifs ont deux fonctions principales: l'entrée des objets graphiques et la désignation des objets graphiques. On distingue normalement 5 sortes de dispositifs d'entrée graphique:

1. les locateurs qui permettent d'entrer une position ou une suite de positions. Les principaux dispositifs de ce type sont:

- la tablette graphique, surface rectangulaire accompagnée d'un crayon qui permet de donner à l'ordinateur la position de ce crayon sur la tablette.

- la souris, dispositif se déplaçant sur roulettes ou glissant sur une surface plane; un ou plusieurs boutons permettent d'entrer une position correspondant à l'emplacement de cette souris

- le "trackball" (Figure 1-3), formée d'une balle que l'on peut faire tourner dans toutes les directions avec la paume de la main pour indiquer les déplacements que l'on souhaite

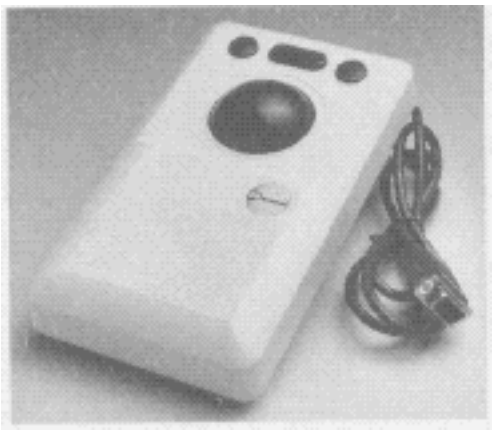

Figure 1-3. Trackball

- le "joystick", sorte de tige que l'on peut aussi mouvoir dans toutes les directions pour indiquer un déplacement

2. **les instruments de désignation** (pick) qui permettent de pointer un objet; le plus connu est le photostyle (light pen, voir Figure 1-4) qui est une sorte de crayon qui détecte la lumière. Ainsi en le déplaçant sur la surface d'un écran, on peut pointer des objets dessinés à l'écran et un signal est envoyé à l'ordinateur, ce qui permet de recueillir les coordonnées de l'objet pointé.

3. **les valuateurs** qui permettent d'entrer une valeur numérique comme un angle, une taille ou un rayon; ce sont typiquement les potentiomètres qui se présentent généralement sous la forme de boutons que l'on peut tourner pour faire varier la valeur choisie.

4. **les claviers** qui permettent d'entrer des objets en tapant des commandes

5. **les boutons** qui permettent de choisir une action parmi un choix donné; la réalisation la plus courante est très certainement sous la forme des clés de fonction dans la plupart des terminaux.

Signalons encore les possibilités d'entrer des images à partir d'une caméra vidéo. Néanmoins, dans ce cas on se trouve confronté à des problèmes d'analyse d'images relevant du domaine du traitement d'images et de la reconnaissance de formes.

Il existe également des dispositifs tridimensionnels très évolués basés sur des capteurs, qui seront présentés en détail dans la cinquième partie consacrée à la réalité virtuelle. Il s'agit notamment des gants (DataGlove, CyberGlove) permettant de communiquer à l'ordinateur toutes les positions des articulations des doigts, des souris 3D, des extensions du "trackball" à 6 degrés de liberté (3 translations et 3 rotations), comme la "SpaceBall". On verra également dans cette cinquième partie les dispositifs stéréos et les casques d'immersions. Il existe aussi des dispositifs permettant d'enregistrer tous les mouvements du corps comme le Flock of Birds d'Ascension ou le Fastrack de Polhemus. Ces dispositifs seront discutés en détail dans la quatrième partie consacrée à l'animation et notamment à la rotoscopie.

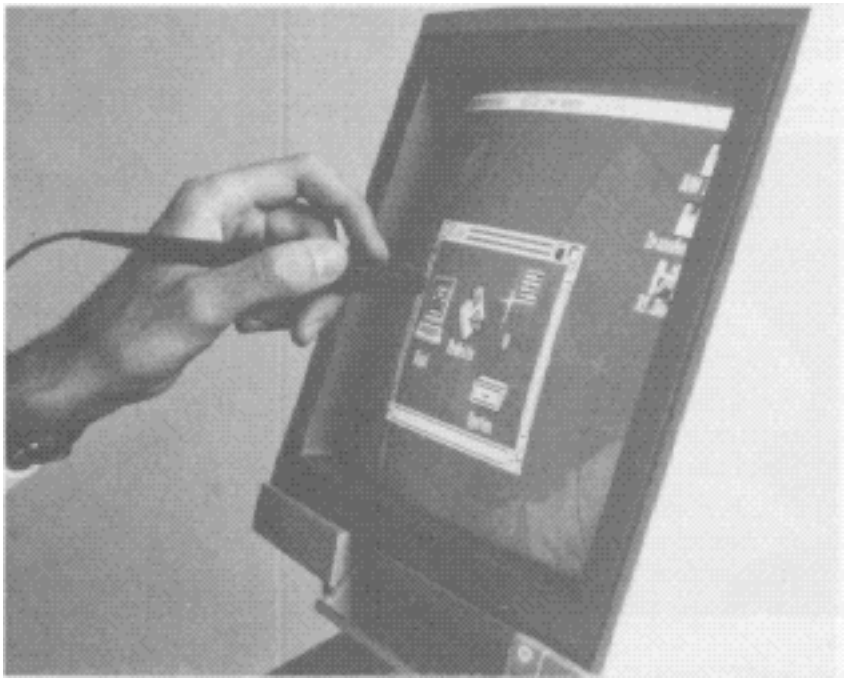

Figure 1-4. Lightpen

1.2.3 Les dispositifs graphiques de sortie

Les principaux dispositifs de sortie sont les écrans de visualisation. Même s'il en existe d'autres variétés comme les écrans à plasma, les deux principaux types sont les écrans calligraphiques et les écrans à balayage récurrent (raster scan).

Dans les écrans calligraphiques, les images sont produites par une suite de segments de droite, ce qui a l'avantage de produire des lignes de très bonne qualité, mais rend difficile le remplissage de polygones. La technologie repose sur les écrans à tube cathodique dont le principe illustré à la Figure 1-5 est le suivant:

Un canon à électrons (émission thermo-électronique) émet un faisceau d'électrons à haute vitesse. Un dispositif de concentration permet d'assurer la convergence du faisceau. Un dispositif de déviation permet de commander la position d'impact sur l'écran. Ce dernier est fait de matériaux

luminescents et fonctionne selon le principe photoélectrique. Les électrons incidents provoquent une excitation des électrons de la matière de l'écran qui en revenant au repos provoque l'émission d'un photon (émission lumineuse).

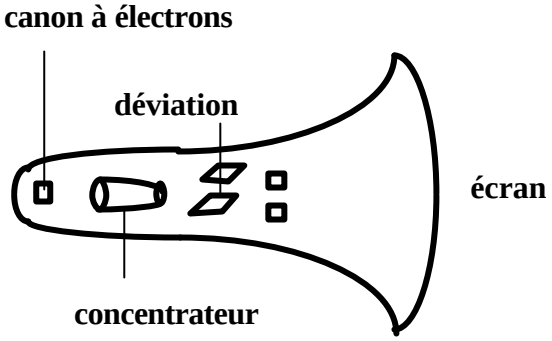

Figure 1-5. Principe du tube cathodique

Les écrans à balayage récurrent, que nous nommerons plus simplement écrans raster, sont proches d'un poste de télévision mais ils sont munis d'une mémoire d'image (frame buffer) qui permet de stocker l'image. Cette mémoire se présente comme une matrice d'informations. La taille de la matrice correspond à la résolution du terminal et chaque information est une élément d'image ou pixel. Pour chaque pixel, on a un nombre de bits, ce qui fixe les possibilités de couleurs pour ce pixel. Par exemple, si on a 8 bits par pixel, on pourra colorier le pixel selon 256 couleurs différentes. En fait, beaucoup de terminaux ont un grand nombre de couleurs à choix, et la valeur d'un pixel est une adresse dans une table de couleurs choisies parmi toutes les couleurs disponibles. Par exemple, la plupart des stations Silicon Graphics ont une résolution de 1280x1024 avec 24 bits pour la couleur, ce qui permet environ 16.7 millions de couleurs différentes.

Le principal défaut des terminaux raster est la mauvaise qualité du tracé de lignes droites. En effet, comme on peut le voir à la Figure 1-6, les droites sont formées d'une suite de pixels, ce qui cause des effets d'escalier ou aliasing. On peut y remédier par des techniques d'antialiasing, mais elles sont souvent coûteuses en temps de traitement.

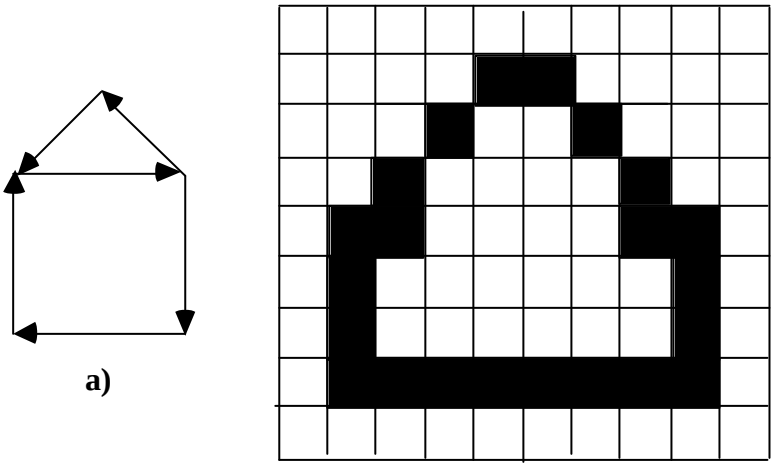

Figure 1-6. Maison a) sur un terminal calligraphique b) sur un terminal raster

1.2.4 Les logiciels graphiques

Ces logiciels peuvent se présenter sous différentes formes:

1. Ensemble de sous-programmes pouvant être "appelés" depuis un langage de programmation comme C; le principal défaut est l'absence de syntaxe dans la construction des objets. Les plus connus à l'heure actuelle sont OPEN-GL et OPEN-INVENTOR.

2. Langages graphiques; aucun n'a eu de réel succès, car cela nécessite d'apprendre un nouveau langage.

3. Extensions de langages généraux: on a l'avantage de profiter de structures existantes et seules les extensions doivent être apprises. Dans cette catégorie, on trouve des extensions de PASCAL (MIRA), C, ADA, EIFFEL, SMALLTALK.

4. Systèmes interactifs: ils regroupent tous les programmes d'application, les systèmes de modélisation et les éditeurs graphiques. On peut citer, par exemple, les logiciels de synthèse d'images de Alias-Wavefront, de Softimage et de StudioMax.

1.3 La modélisation des objets

1.3.1 2D versus 3D

L'être humain vit dans un monde à trois dimensions, mais lorsqu'il dessine, il utilise généralement des feuilles de papier qui n'ont que deux dimensions. Il se trouve donc confronté à un problème de représentation en deux dimensions d'un monde à trois. Deux solutions s'offrent alors:

1. représenter seulement une face plane des objets, par exemple la façade avant d'une maison ou le dessus d'une table

2. tenter de dessiner la scène choisie en tenant compte de lois de projection telle que la perspective.

En informatique graphique, comme les supports matériels (écrans) sont à deux dimensions, ces deux approches se retrouvent et donnent lieu à deux types de modélisation, de systèmes graphiques et d'applications.

On dira qu'un système graphique est à deux dimensions (2D) si la représentation interne de l'information graphique dans l'ordinateur est à deux dimensions. Un système graphique sera à trois dimensions (3D) lorsque l'ordinateur a connaissance de l'information tridimensionnelle. Cette distinction est fondamentale. En effet, lorsqu'on voit une image produite par ordinateur d'une maison en perspective, il est impossible de savoir si l'image a été produite avec un système à 2 ou à 3 dimensions. En effet, la maison a pu être dessinée en perspective et fournie ainsi à un système graphique à 2 dimensions qui s'est contenté de la restituer ou la vue en perspective a été synthétisée par un système tridimensionnel à partir de données tridimensionnelles. Ceci nous amène à préciser que lorsque nous parlerons d'images tridimensionnelles, il s'agira toujours d'images produites à partir d'un modèle tridimensionnel connu de l'ordinateur et non d'images réellement en trois dimensions, telles que celles produites par des techniques comme l'holographie ou la stéréoscopie.

Il faut aussi remarquer que l'espace à deux dimensions peut être considéré comme un cas particulier d'espace à trois dimensions dont la troisième dimension Z est toujours nulle. Pour cette raison, nous conviendrons de présenter la plupart des notions dans l'espace tridimensionnel. Nous choisirons également des systèmes de coordonnées tels que la troisième dimension puisse simplement s'ajouter comme le montre la Figure 1-7.

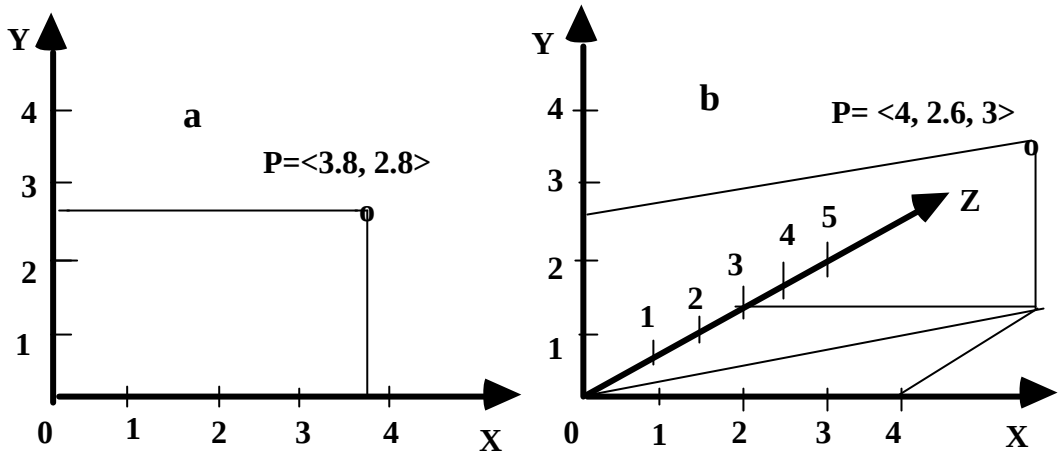

Figure 1-7. Systèmes de coordonnées a) 2D b) 3D

1.3.2 Points et vecteurs

L'objet graphique le plus simple est évidemment le point caractérisé par ses coordonnées et noté P = $<P_x,P_y,P_z>$. Des exemples sont montrés à la Figure 1-7. Nous utiliserons une notation semblable pour représenter les vecteurs qui jouent un rôle fondamental en informatique graphique. Un vecteur sera considéré comme la direction donnée par la flèche reliant l'origine du système d'axes au point donné par les composantes du vecteur (Figure 1-8).

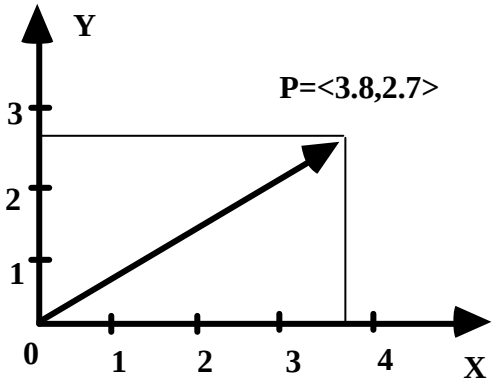

Figure 1-8. Représentation d'un vecteur

Nous noterons un certain nombre d'opérations importantes sur les vecteurs:

norme : $$|V| = \sqrt{V_x^2 + V_y^2 + V_z^2}$$

addition : $V_1 + V_2 = <V_{1x}+V_{2x}, V_{1y}+V_{2y}, V_{1z}+V_{2z}>$

produit scalaire : $V_1 \bullet V_2 = \|V_1\| \|V_2\| \cos \alpha$

où α est l'angle entre les 2 vecteurs il faut noter que le résultat est un nombre réel et que dans un système orthonormé, on a :

$$V_1 \bullet V_2 = V_{1x}V_{2x}+V_{1y}V_{2y}+V_{1z}V_{2z}$$

ce qui permet de déduire l'angle entre les vecteurs.

produit vectoriel:

$V_1 \times V_2 = <V_{1y}V_{2z}-V_{1z}V_{2y}, V_{1z}V_{2x}-V_{1x}V_{2z}, V_{1x}V_{2y} - V_{1y}V_{2x}>$

L'intérêt principal du produit vectoriel est qu'il fournit un vecteur perpendiculaire au plan des deux vecteurs intervenant dans le produit. On peut encore noter que

$$\|V1 \times V2\| = \|V1\| \|V2\| \sin \alpha$$

où α est l'angle entre les 2 vecteurs V_1 et V_2.

1.3.3 Droites, segments de droite et modélisation en lignes

La droite est une figure très courante bien que l'on utilise plutôt le segment de droite. La différence est simple, une droite passe par deux points, tandis qu'un segment de droite est limité par 2 points. Pour tracer un segment de droite AB, nous utiliserons deux instructions, une pour se positionner au point A et une pour tracer le segment de A à B:

```
moveabs A;
lineabs B
```

Pour tracer la maison de la Figure 1-9, on écrira donc:

```
moveabs <<1,6>>;
lineabs<<9,6>>,<<5,7>>,<<1,6>>,<<1,1>>,<<9,1>>,<<9,6>>;
moveabs <<4,1>>;
lineabs <<4,3>>,<<6,3>>,<<6,1>>;
```

Le défaut d'une telle approche est l'utilisation de coordonnées absolues, on préfère donc souvent les coordonnées relatives. On fixe une origine du dessin (le premier point) et tous les déplacements sont relatifs à cette origine. Notre exemple devient alors.

```
moveabs <<1,6>>;
linerel <<8,0>>,<<-4,1>>,<<-4,1>>,<<0,-5>>,<<8,0>>,<<0,5>>;
moverel <<-5,5>>;
linerel <<0,2>>,<<2,0>>,<<0,-2>>
```

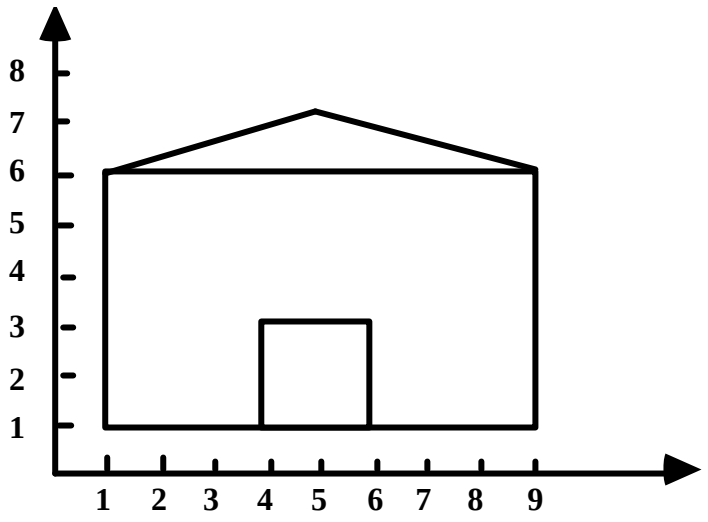

Figure 1-9. Une maison

Ce style de programmation peut suffire à réaliser la plupart des dessins en ligne simples et à deux dimensions. En fait, les opérations **lineabs**, **moveabs**, **moverel** et **linerel** sont très primitives, mais présentes dans la plupart des langages, notamment sur micro-ordinateur (BASIC). On peut d'ailleurs construire des abstractions basées sur ces simples instructions. Par exemple, notre maison peut être paramétrisée et construite comme un type graphique de haut niveau. En se basant sur le dessin de la Figure 1-10, on obtient la programmation suivante:

```
type MAISON=figure (REF: VECTOR; HAUTE, LARGE, HTOIT, HPORTE, LPORTE:REAL);
    var DEMI:REAL;
    begin
       DEMI:=LARGE/2;
       moveabs REF;
       linerel <<LARGE,0>>,<<-DEMI,HTOIT>>,<<-DEMI,-HTOIT>>, <<0,HAUTE>>, <<LARGE,0>>,
                  <<0,HAUTE>>;
       moverel <<-DEMI-LPORTE/2,-HAUTE>>;
       linerel <<0,HPORTE>>,<<LPORTE,0>>,<<0,-HPORTE>>
    end;
```

On peut alors aisément définir deux maisons et les créer avec des dimensions différentes:

```
var M1, M2 : MAISON

          create M1 (<<1,6>>,5,8,1,2,2);
          create M2 (<<10,4>>,2.5,6,2,1.25,1.75)
```

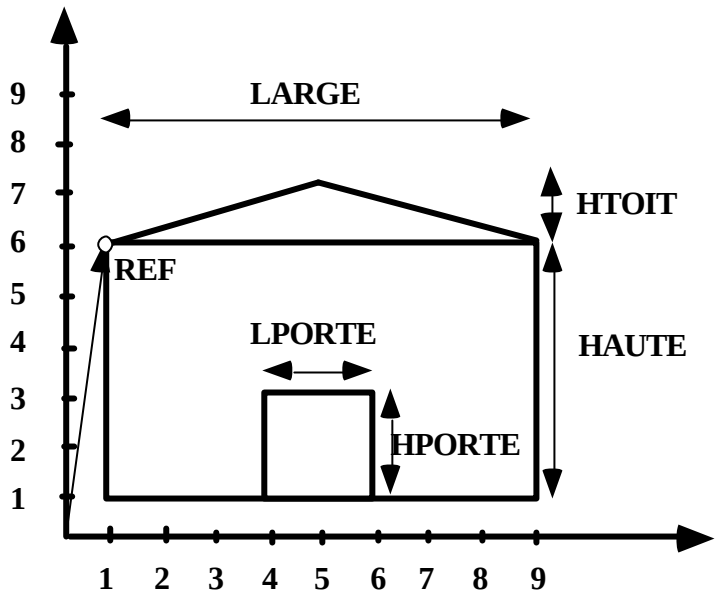

Figure 1-10. Maison paramétrisée

1.4 Le concept de tortue

Le concept de tortue a été introduit dans le cadre du langage LOGO par Seymour Papert du MIT pour apprendre la programmation aux enfants. La tortue est en fait une entité se déplaçant avec son propre système d'axes. Les instructions de commandes à cette tortue sont donc relatives à sa position et son orientation. On tourne à droite par rapport à la direction courante et non par rapport à un référentiel fixe. Les instructions de la tortue les plus usuelles sont les suivantes:

ShowTurtle	rend la tortue visible
HideTurtle	cache la tortue
Forward A	déplace la tortue en avant de A pas
Back A	déplace la tortue en arrière de A pas
Right α	tourne à droite de l'angle α
Left α	tourne à gauche de l'angle α
SetPos P	positionne la tortue au point P
Position	retourne la position courante de la tortue
Home	place la tortue en <0, 0>

Pour tracer un carré de côté 6, on peut faire:

```
ShowTurtle
Forward 6
Right 90
```

```
Forward 6
Right 90
Forward 6
Right 90
Forward 6
```

ou avec une boucle: repeat 4 [Forward 6 Right 90]

Il est aussi possible de définir des procédures comme la procédure CARRE avec un paramètre :cote qui est la longueur du côté. La procédure aura la forme:

```
To CARRE :cote
      repeat 4 [Forward :cote Right 90]
End
```

Pour dessiner un carré de côté 6, on appelera donc: CARRE 6

Ce concept de tortue a été étendu à trois dimensions dans le cadre du langage d'animation ASAS et des L-systèmes (voir Section 2.6). L'orientation courante de la tortue est représentée par trois vecteurs $\vec{H}$, $\vec{L}$, $\vec{U}$ indiquant la direction de la tortue, la direction de la gauche et la direction du haut. Le système est orthonormé selon l'équation:

$$\vec{H} \times \vec{L} = \vec{U}$$

Six rotations sont alors possibles: une rotation gauche et une rotation droite autour de chacun des 3 axes. On trouvera plus de détails à la Section 2.6.3.

1.4.1 Modélisation en polygones

1.4.1.1 Le concept de polygone

Qu'on travaille en deux dimensions ou en trois, le polygone joue un rôle extrèmement important. Sa définition n'est pas toujours rigoureuse et varie selon les auteurs. Nous entendons par polygone une figure plane definie par une liste de points (les sommets) reliés par des segments de droite (les arêtes). Les sommets sont supposés tous différents, les arêtes ne doivent pas se croiser et une arête relie le dernier sommet au premier. Un polygone est concave s'il existe au moins un angle interne supérieur à 180^o; il est convexe, s'il n'est pas concave.

A deux dimensions, les polygones sont utilisés car en les remplissant de couleurs, on construit rapidement des images attrayantes.

1.4.1.2 Algorithme de remplissage

L'algorithme de base le plus populaire pour remplir un polygone consiste à balayer le polygone par des lignes horizontales (lignes de balayage). A chaque ligne, on répète les étapes suivantes:

1. trouver les intersections de la ligne de balayage avec toutes les arêtes du polygone

2. trier ces intersections dans l'ordre croissant des coordonnées x

3. mettre à la valeur donnée tous les pixels situés entre les paires d'intersection.

La Figure 1-11 montre un exemple; il faut noter qu'à la ligne de balayage 6, on doit prendre garde au sommet double S5.

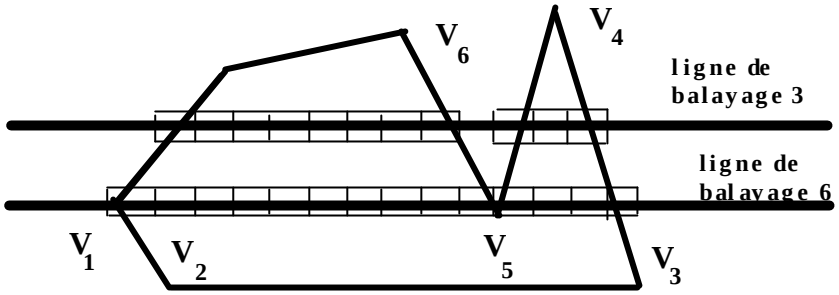

Figure 1-11. Remplissage par balayage

Bien qu'il soit souvent coûteux en temps d'exécution, le modèle de représentation d'objets tridimensionnels par polygones est le plus courant. Dans ce modèle tous les objets sont découpés en faces polygonales. On peut par exemple construire une tête à partir d'un buste (Figure 1-12) sur lequel on dessine les polygones (Figure 1-13), avant de les numériser.

Figure 1-12. Buste servant à la numérisation

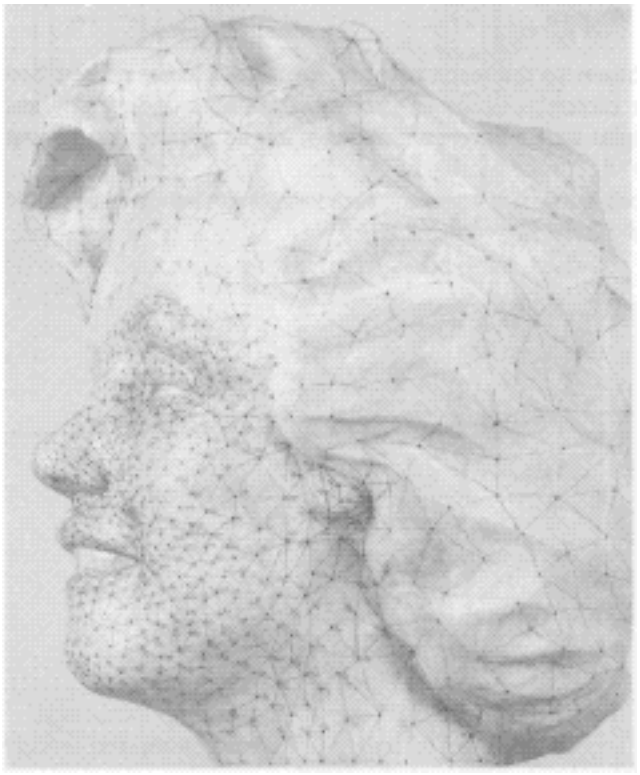

Figure 1-13. Les polygones sont directement dessinés sur le buste

On obtient alors un modèle que l'on peut représenter en fil de fer ou en surface (Figure 1-14). Pour des objets tels que des cubes ou des polyèdres réguliers, le découpage en polygones est évidemment approprié, pour des objets tels que des sphères, des surfaces de révolution, ou un visage, on est obligé de procéder à des approximations.

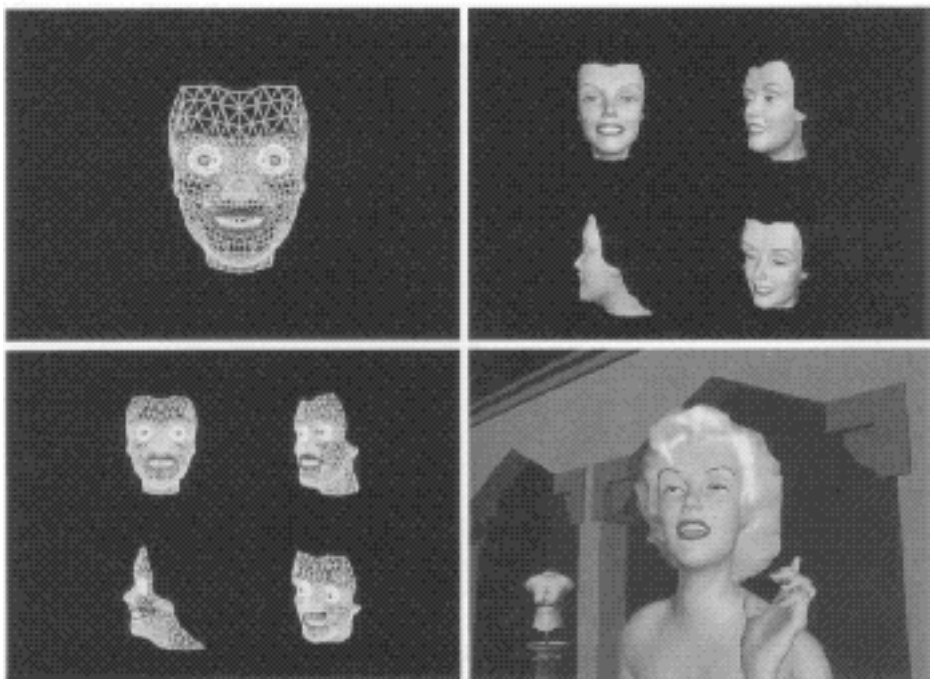

Figure 1-14. Le visage obtenu en fil de fer et avec un rendu réaliste

Pour construire un objet modélisé par polygones, le principe est toujours le même, il faut une liste de sommets et une liste de faces avec pour chacune les références aux sommets de la face dans la liste globale. Pour illustrer le modèle, nous allons prendre l'exemple d'une boite (parallélépipède) définie par 4 sommets A,B,C et D. L'exemple est implanté en langage MIRA et correspond à la boite de la Figure 1-15.

```
type BOITE=figure(A,B,C,D:VECTOR);
var CORI, BORI, DORI: VECTOR;
begin
  CORI:=C-A; BORI:=B-A; DORI:=D-A;
  vertices:=A,C,B+CORI,B,D,C+DORI,D+BORI+CORI,D+BORI;
  createface 1 to 6 with 4 edges;
  face 1:=1,2,3,4;  face 2:=2,6,7,3;  face 3:=3,7,8,4;
  face 4:=5,1,4,8;  face 5:=1,5,6,2;  face 6:=6,5,8,7
end;
```

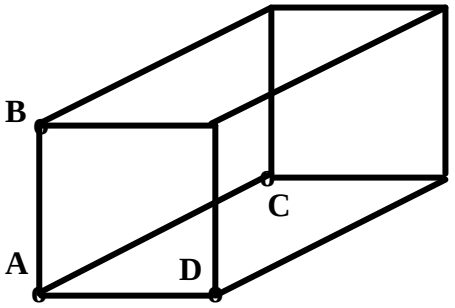

Figure 1-15. Une boite

1.4.2 Modèles internes et externes

Un modèle interne est un modèle caché à l'usager; c'est une manière de stocker l'information. Un modèle externe est un modèle pour l'usager, un tel modèle est souvent caractérisé par des paramètres et on utilise souvent un modèle procédural. Par exemple, un cube peut-être défini par son centre, le centre d'une face et la direction vers un des sommets de cette face; c'est un modèle externe du cube. Le cube peut être représenté dans la machine selon deux modèles internes différents:

a) comme une suite de déplacements (modèle en lignes)

b) comme une série de facettes polygonales

On peut encore considérer un modèle d'affichage; par exemple, le cube représenté par facettes peut être dessiné en lignes ou en facettes. On a donc le schéma de la Figure 1-16. Attention: le passage du modèle interne lignes au modèle d'affichage facettes est quasiment impossible. En pratique, il faut noter que l'on utilise souvent le même modèle (par exemple, facettes) pour les trois niveaux externe, interne et d'affichage.

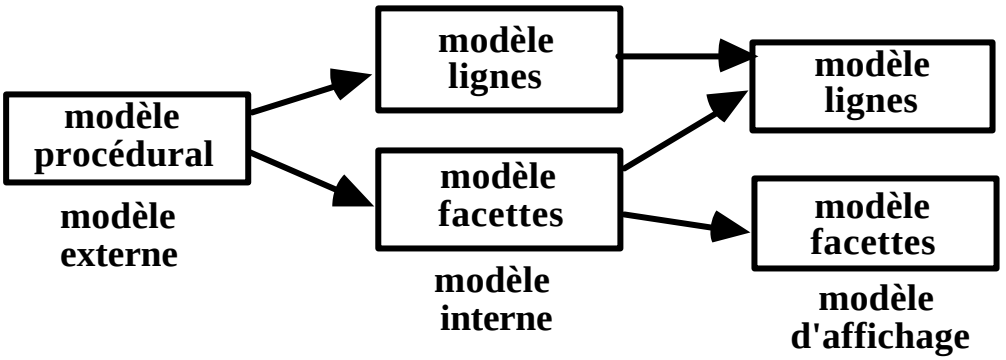

Figure 1-16. Les relations entre les modèles

1.5 Les transformations ponctuelles

1.5.1 Les transformations de base

Une fois que l'on a construit des objets graphiques, on désire généralement les manipuler, c'est-à-dire changer leurs attributs. Considérons donc les attributs d'un objet graphique, on peut relever:

- la position
- l'orientation
- la taille
- la forme
- la couleur
- la transparence
- la texture

La forme est un attribut particulier et sa modification peut s'avérer très compliquée. La couleur sera traitée à la Section 1.7 et la transparence et la texture sont des attributs seulement présents dans les images réalistes traitées dans la partie 2. Les trois premiers attributs ont en commun qu'ils peuvent être modifiés par des transformations dites ponctuelles. Ainsi:

- on modifie la position par des translations
- on modifie l'orientation par des rotations
- on modifie la taille par des transformations d'échelle

Ces transformations sont ponctuelles car elles s'appliquent sur tout point P de l'objet pour donner un nouveau point P'. On peut donc définir chaque transformation par la relation permettant de passer de $P = <P_x,P_y,P_z>$ à $P' = <P'_x,P'_y,P'_z>$. Ainsi, on a:

translation d'un vecteur $T = <T_x,T_y,T_z>$

On additionne les composantes de T aux coordonnées de P:

$P' = P + T = <P_x+T_x, P_y+T_y, P_z+T_z>$

rotation d'un angle A autour d'un des axes

La rotation est possible autour de chacun des axes:

autour de **x**: P' = <P_x, P_y*cos A - P_z*sin A, P_y*sin A + P_z*cos A>

autour de **y**: P' = <P_x*cos A + P_z*sin A, P_y, -P_x*sin A + P_z*cos A>

autour de **z**: P' = <P_x*cos A - P_y*sin A, P_x*sin A + Py*cos A, P_z>

transformation d'échelle d'un facteur E = <E_x,E_y,E_z>

Cette transformation revient à effectuer le produit de P par le vecteur E:

P' = P*E = <P_x*E_x, P_y*E_y, P_z*E_z>

Plutôt que d'employer ces équations, on préfère utiliser une notation matricielle. Mais s'il est aisé de définir des matrices de rotation et de transformation d'échelle, pour intégrer la translation, il faut introduire le concept de coordonnées homogènes.

1.5.2 Coordonnées homogènes (4D)

Avec ces coordonnées, on travaille dans un espace à 4 dimensions où chaque point P est défini comme P = <P_x, P_y, P_z, 1>. Inversément, tout point de coordonnées <x,y,z,w> dans l'espace 4D représente le point <x/w, y/w, z/w> dans l'espace 3D. Dans cette représentation 4D, les transfomations ponctuelles s'expriment par des matrices 4x4.

translation d'un vecteur T:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix}$$

rotation d'un angle A autour de l'axe X:

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotation d'un angle B autour de l'axe Y:

$$R_y = \begin{bmatrix} \cos b & 0 & \sin b & 0 \\ 0 & 1 & 0 & 0 \\ -\sin b & 0 & \cos b & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

rotation d'un angle C autour de l'axe Z:

$$R_z = \begin{bmatrix} \cos g & -\sin g & 0 & 0 \\ \sin g & \cos g & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

transformation d'échelle d'un facteur E:

$$E = \begin{bmatrix} E_x & 0 & 0 & 0 \\ 0 & E_y & 0 & 0 \\ 0 & 0 & E_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

1.5.3 Concaténation de transformations

Pour effectuer plusieurs transformations ponctuelles de suite, il suffit de multiplier les matrices de transformations et d'appliquer la matrice résultante comme matrice de transformation globale. En effet, on a:

$$P' = P*T1*T2 = P*(T1*T2)$$

Prenons par exemple, la rotation d'un angle γ autour d'une droite arbitraire d, donnée par un point P et un vecteur directeur unitaire N.

Les étapes de réalisation sont les suivantes:

1. Translation de -P pour amener d à passer par l'origine

2. Rotation de α autour de l'axe y, puis de β autour de l'axe x pour amener la droite d à coïncider avec l'axe z. En observant la Figure 1-17, on peut en déduire les matrices suivantes:

$$R_{y\alpha} = \begin{bmatrix} \frac{Nz}{Nxz} & 0 & -\frac{Nx}{Nxz} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{Nx}{Nxz} & 0 & \frac{Nz}{Nxz} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_{x\beta} = \begin{bmatrix} Nxz & 0 & -Ny & 0 \\ 0 & 1 & 0 & 0 \\ Ny & 0 & Nxz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

où N_{xz} est la projection de N sur le plan xz.

3. Rotation autour de l'axe z de l'angle donné γ
4. Rotation inverse de $R_{x\beta}$ et rotation inverse de $R_{y\alpha}$
5. Translation de P

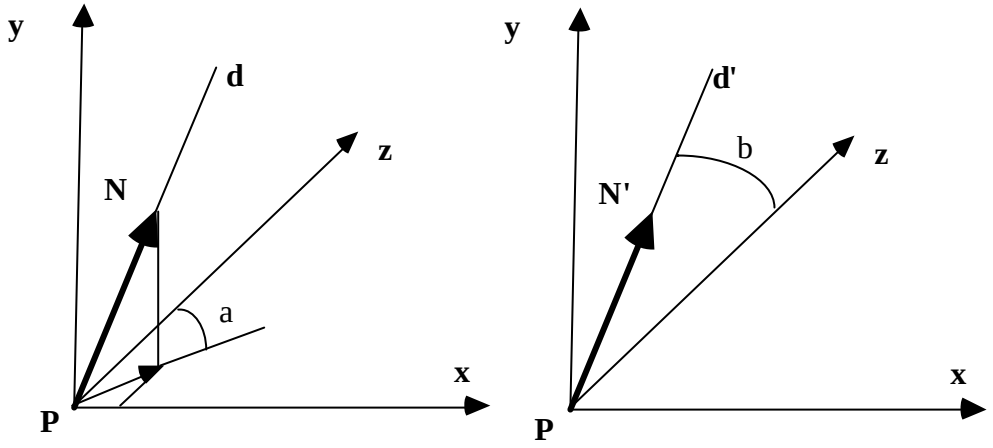

Figure 1-17. Rotation pour faire coïncider une droite d avec l'axe z

En remarquant que les transformations inverses s'expriment aisément:

$$T_P^{-1} = T_{-P}$$

$$R_\alpha^{-1} = R_{-\alpha}$$

$$E_S^{-1} = E_{<1/Sx,1/Sy,1/Sz>}$$

on obtient pour la matrice de rotation générale:

$$R\gamma = T_P * R_{y\alpha} * R_{x\beta} * R_{zC} * R_{x(-\beta)} * R_{y(-\alpha)} * T_{-P}$$

1.5.4 Autres transformations

Parmi les autres transformations ponctuelles, on peut noter les symétries et l'homothétie. Les symétries par rapport aux trois plans principaux sont fréquemment utilisées. On peut les exprimer en coordonnées homogènes par les matrices suivantes:

$$S_{xy} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad S_{yz} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad S_{xz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

L'homothétie est une opération bien connue en optique géométrique qui multiplie toutes les distances par un facteur R (rapport d'homothétie) à partir d'un point (centre d'homothétie). La Figure 1-18 nous montre un exemple. La matrice d'homothétie peut être obtenue en effectuant une translation de -C, puis une transformation d'échelle de <R,R,R> puis une translation de C. Donc: $H_{RC} = T_{-C} * E_{<R,R,R>} * T_C$

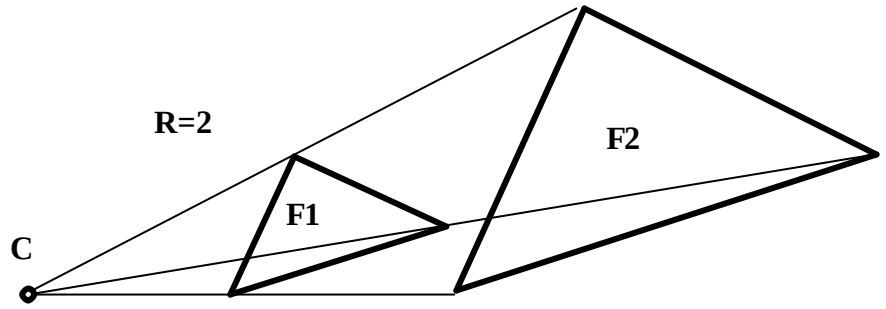

Figure 1-18. Exemple d'homothétie

1.6 Les transformations visuelles

1.6.1 Fenêtre et clôture

Même si on est capable de modéliser un objet dans l'ordinateur et de le transformer, il n'en reste pas moins que pour le voir sur un écran graphique, il faut passer de l'espace de l'usager (généralement tridimensionnel) à l'espace de l'écran. Non seulement ce dernier est à deux dimensions, mais les constructeurs de matériel graphique se sont ingéniés à définir des espaces d'adresses compliqués et très différents d'un modèle à l'autre. Le but des transformations visuelles est donc de passer de l'espace de l'usager à celui du dispositif graphique. Nous commencerons par le cas simple d'un monde de l'usager à deux dimensions. Ceci nous amène à définir deux concepts fondamentaux: la fenêtre et la clôture.

Pour ne pas s'occuper de l'espace d'adresses d'un terminal particulier. nous allons considérer que l'écran se représente par un rectangle dont le sommet inférieur gauche est à <0,0> et le sommet supérieur droit est à <1,1>. Ainsi toute portion de l'écran que nous utiliserons sera nécessairement limitée par des vecteurs de composantes comprises entre 0 et 1. Cette portion du dispositif graphique est appelée la *clôture*.

Notre monde réel, supposé à deux dimensions, est évidemment illimité et nous devons spécifier quelle portion du monde réel nous voulons représenter. Cette portion est un rectangle, nommé *fenêtre*, et nous le donnerons par son sommet inférieur gauche et son sommet supérieur droit. C'est ainsi que le contenu de la fenêtre sera toujours représenté dans la clôture. En considérant la Figure 1-19, on peut montrer comment on passe d'un point P dans la fenêtre à un point P' dans la clôture:

$$P'_x = (CH_x - CB_x) * (P_x - FB_x) / (FH_x - FB_x) + Cb_x$$

$$P'_y = (CH_y - CB_y) * (P_y - FB_y) / (FH_y - FB_y) + Cb_y$$

1.6.2 Coupage selon la fenêtre

Nous avons encore un problème à résoudre: comment couper les objets qui sortent de la fenêtre ? Il s'agit, dans le cas de dessins en lignes, de couper chaque segment selon les bords de la fenêtre comme le montre la Figure 1-20.

La méthode la plus connue pour effectuer cette opération est la méthode de Cohen-Sutherland. Cet algorithme est basé sur une division du plan en 9 régions dont la région centrale est la fenêtre. La Figure 1-21 nous montre cette division.

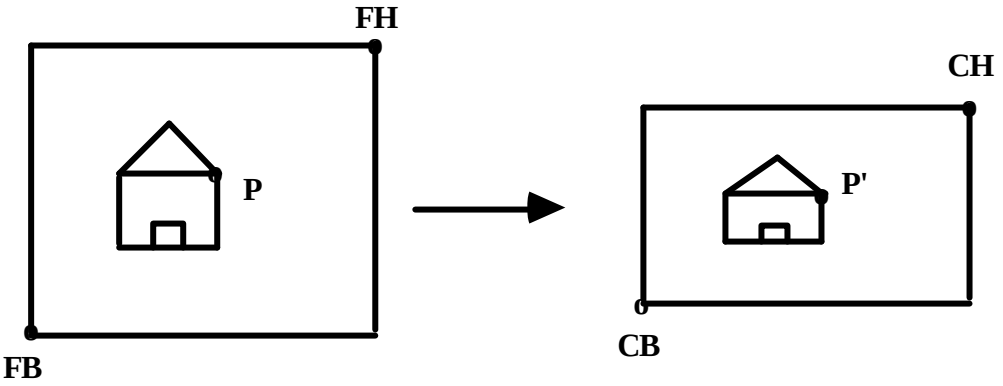

Figure 1-19. Passage de la fenêtre à la clôture

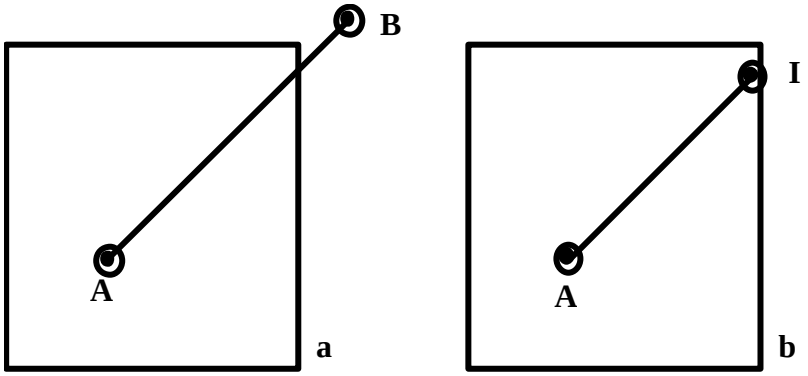

Figure 1-20. Coupage d'un segment a) avant b) après

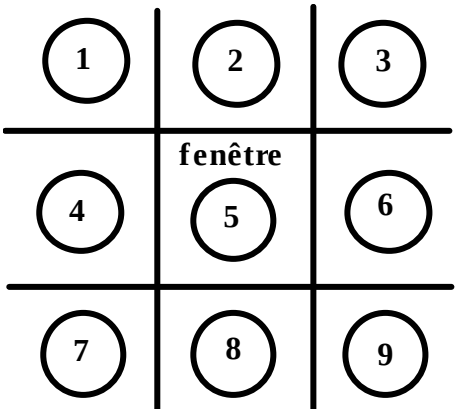

Figure 1-21. Les 9 régions dans l'algorithme de Cohen-Sutherland

Avec cette approche, on élimine rapidement des cas simples comme des segments dont les extrémités sont dans les régions 1 et 3 ou 6 et 9. Pour les segments qui traversent la fenêtre, on calcule successivement les intersections des segments avec les lignes bordant la fenêtre, comme on le voit dans l'exemple de la Figure 1-22.

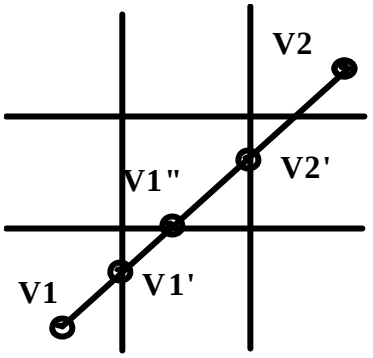

Figure 1-22. Principe de l'algorithme de Cohen-Sutherland

1.6.3 Projections parallèles et perspectives

Revenons maintenant au cas général de l'espace à trois dimensions. Le problème principal est de passer de trois à deux dimensions, ce qui suppose la projection des objets de l'espace sur un plan, que l'on appelle le plan de vue. On distingue deux grands types de projections: la projection parallèle (Figure 1-23a) et la perspective (Figure 1-23b). dans les deux cas, la projection de l'objet s'obtient en faisant passer des droites (les projecteurs) par chaque point de l'objet et en cherchant l'intersection avec le plan de vue. Dans le cas d'une perspective, toutes les lignes émanent d'un seul point : le centre de projection. Pour une projection parallèle, toutes les droites sont parallèles à une direction de projection. Si la direction de projection est perpendiculaire au plan de vue, on a une projection orthographique sinon elle est oblique.

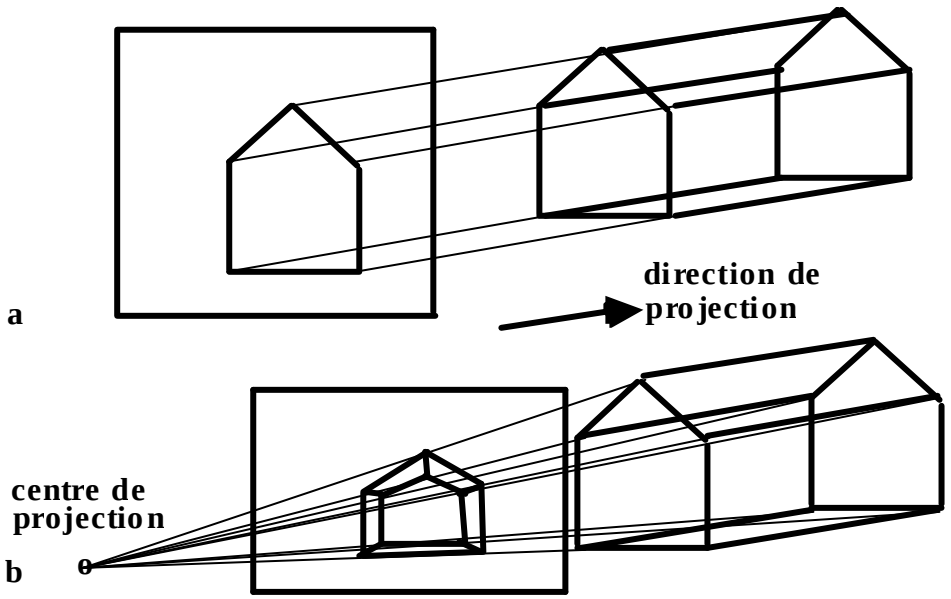

Figure 1-23. a) principe de la projection parallèle b) principe de la perspective

On va définir la fenêtre comme le rectangle délimitant ce qui doit être vu dans le plan de vue. La fenêtre est spécifiée selon un système d'axes UVW où V est la projection sur le plan de vue d'un vecteur particulier appelé le vecteur viewup et W le vecteur normal au plan de vue. L'axe U est alors à 90^0 comme on le voit dans la Figure 1-24.

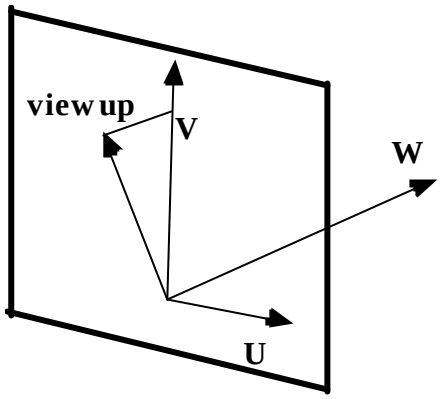

Figure 1-24. Système d'axes de la fenêtre

On définira aussi un volume de vue qui contiendra tous les objets susceptibles d'être projetés. Dans le cas d'une perspective, le volume de vue est une pyramide semi-infinie dont le sommet est le centre de projection et la fenêtre une section. Dans le cadre d'une projection parallèle, le volume de vue est un parallélépipède.

Nous pouvons maintenant décrire les étapes de traitement d'une scène tridimensionnelle:

1. Coupage selon le volume de vue
2. Projection sur la fenêtre dans le plan de vue
3. Passage de la fenêtre à la clôture
4. Passage de la clôture aux coordonnées du dispositif graphique

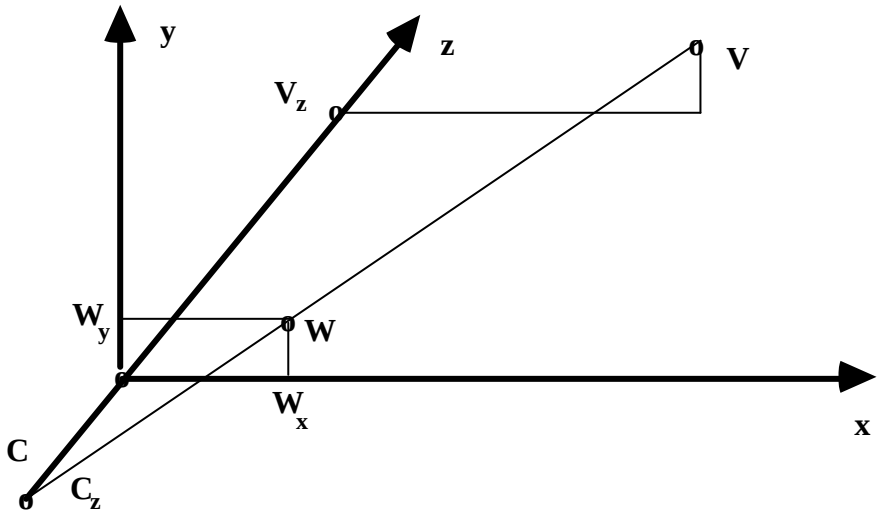

Figure 1-25. Perspective d'un point V

Nous n'allons pas traiter en détails toutes ces transformations, mais nous allons montrer comment déterminer la matrice de perspective dans un cas très simple. On suppose donc que le centre de

projection est placé sur la partie négative de l'axe Z, le plan de vue est sur le plan XY. On considère un point V et sa projection W. La Figure 1-25 nous montre la situation.

On voit aisément qu'on a les relations suivantes:

$$W_x = V_x/(1-V_z/C_z) \quad \text{et} \quad W_y = V_y/(1-V_z/C_z)$$

Or si on applique la matrice suivante :

$$M_{per} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{C_z} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

sur le vecteur 4D $V=<V_x,V_y,V_z,1>$, on obtient $V' = <V_x,V_y,0,1-V_z/C_z>$ en ramenant à trois dimensions, comme on l'a vu dans la Section 1.5.2, on obtient $<V_x/(1-V_z/C_z), V_y/(1-V_z/C_z, 0>$ ce qui est W. Donc M_{per} est la matrice de perspective et plus généralement, on peut affirmer que toute matrice de coordonnées homogènes (4x4) qui a des termes non nuls dans les trois premiers éléments de la quatrième colonne est une matrice de perspective.

Pour terminer, il faut signaler que pour l'usager d'un système tridimensionnel, on lui donne généralement une spécification très simple sous forme d'ine caméra virtuelle comprenant:

- un oeil, c'est-à-dire le point d'où l'on regarde
- un point d'intérêt, c'est-à-dire le point que l'on regarde
- un angle de vue (45^o pour un être humain moyen)

Le système va alors calculer la matrice de perspective correspondante.

1.7 La couleur

1.7.1 Le rôle de la couleur en informatique graphique

En informatique graphique, la couleur joue un rôle fondamental pour deux raisons principales. D'une part, elle permet de distinguer des objets différents, d'autre part, elle est indispensable pour rendre les images réalistes. Pour distinguer N objets différents, il suffit de N couleurs et souvent moins; comme on manipule rarement beaucoup d'objets à la fois, on peut donc se contenter d'un nombre limité de couleurs. Dans le cas d'images réalistes, on fait intervenir la lumière; mais, si on considère, par exemple, un objet complexe rouge, il faudra un grand nombre de nuances rouges pour le représenter. Plus généralement, la production d'images réalistes avec transparence, texture, ombrage nécessite une très grande quantité de couleurs.

On constate donc que selon le type d'applications envisagées, le besoin en couleurs est variable. Il faut pourtant des moyens standards de spécifier ces couleurs. Avec un nombre limité à 8 par exemple, on peut évidemment se servir des noms des couleurs; mais pour des milliers voire des

millions de couleurs, des systèmes numériques sont indispensables. Il existe plusieurs systèmes dont les principaux sont : RGB, CMY, YIQ, HSV et HLS. Nous ne décrirons que les systèmes RGB et HLS.

1.7.2 Le système RGB

Ce système est dérivé du système instauré en 1931 par la Commission Internationale de l'Eclairage (CIE). Le système de la CIE définissait un espace basé sur trois couleurs primaires: Rouge (**R**ed), Vert (**G**reen) et Bleu (**B**lue). Toute couleur visible devenait une combinaison linéaire des trois couleurs primaires. Le système RGB correspond au principe des moniteurs TV, puisque dans ces moniteurs, les couleurs sont crées par des phosphores rouges, verts et bleus.

Le modèle RGB utilise généralement un cube dont les arêtes sont de longueur 1 et qui est représenté à la Figure 1-26. Le noir est à l'origine et le blanc au point <1,1,1>. Les trois couleurs primaires se trouvent évidemment le long des trois axes. Enfin, les couleurs cyan, magenta et jaune sont situées aux trois autres sommets du cube. Toutes les couleurs peuvent ainsi être exprimées par des vecteurs de composantes comprises entre 0 et 1. On aura donc:

Rouge = <1,0,0> Vert = <0,1,0> Bleu=<0,0,1>

Jaune = <1,1,0> Magenta = <1,0,1> Cyan = <0,1,1>

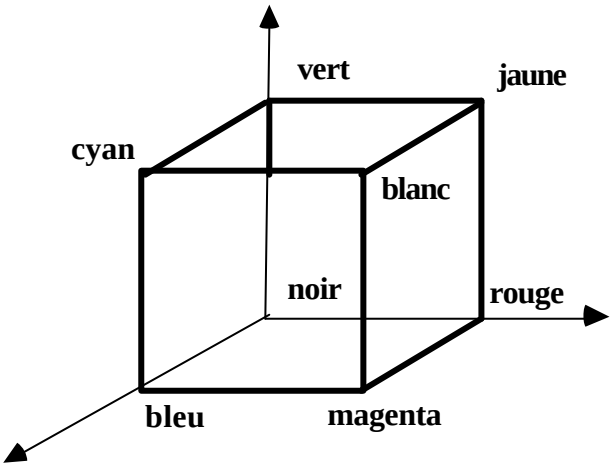

Figure 1-26. Le cube RGB

Il faut encore remarquer que le long de la diagonale du cube, on trouve une échelle des gris allant du noir (<0,0,0>) au blanc (<1,1,1>).

1.7.3 Le système HLS

Ce système spécifie aussi toute couleur à l'aide de trois nombres, mais ces trois nombres ont une toute autre signification. Ce sont la nuance ou teinte (**H**ue), la clarté (**L**ightness) et la saturation (**S**aturation).

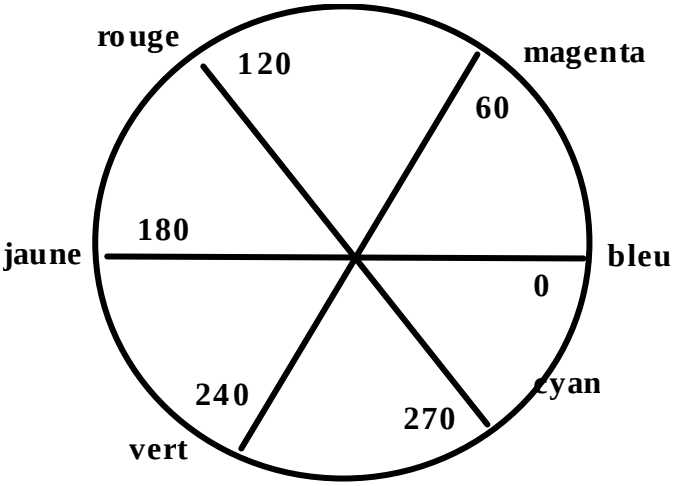

Figure 1-27. Le cercle des teintes dans le système HLS

La nuance ou teinte H peut se représenter à l'aide d'un cercle; on considère alors l'angle au centre du cercle. En degrés, la teinte va donc de 0^o à 360^o comme le montre la Figure 1-27. Les trois couleurs primaires et les trois couleurs complémentaires forment un héxagone régulier.

La clarté L est définie selon une échelle allant de 0 (noir) à 1 (blanc) en passant par tous les gris.

La saturation S mesure la pureté des couleurs, c'est-à-dire le pourcentage de couleur pure par rapport au blanc. Une valeur de 1 représente donc une couleur pure ou saturée tandis qu'une valeur de 0 correspond à un gris de même clarté.

Le système HLS peut être illustré à l'aide d'un double cône (Figure 1-28). A la surface du cône, toutes les couleurs ont une saturation de 1. La saturation est représentée le long du rayon d'une section circulaire du cone. La teinte est décrite par l'angle au centre d'un cercle tandis que la clarté est sur l'axe vertical.

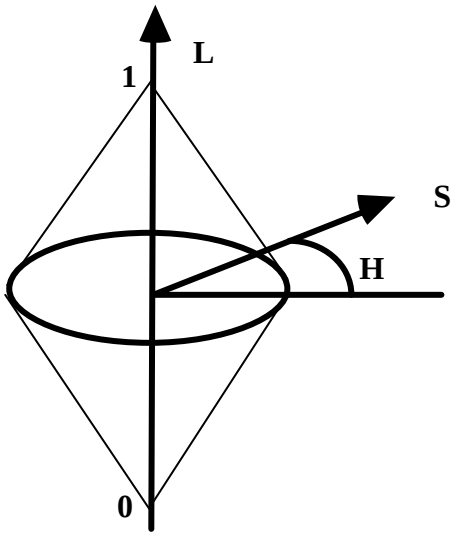

Figure 1-28. Le double cône HLS

Remarquons que la teinte est souvent exprimée en fraction de tour de cercle, ce qui permet de noter les couleurs par des vecteurs à trois composantes comprises entre 0 et 1: <H,L,S>. Ainsi, les couleurs primaires et complémentaires peuvent s'écrire:

Rouge = <0.33,0.5,1> Vert = <0.67,0.5,1> Bleu=<0,0.5,1>

Jaune = <0.5,0.5,1> Magenta = <0.167,0.5,1> Cyan = <0.833,0.5,1>